

# Fast orthogonal transforms for multi-level quasi-Monte Carlo integration

Christian Irrgeher\* and Gunther Leobacher†

August 11, 2015

## Abstract

We combine a generic method for finding fast orthogonal transforms for a given quasi-Monte Carlo integration problem with the multilevel Monte Carlo method. It is shown by example that this combined method can vastly improve the efficiency of quasi-Monte Carlo.

## 1 Introduction

Many simulation problems from finance and other applied fields can be written in the form  $\mathbb{E}(f(X))$ , where  $f$  is a measurable function on  $\mathbb{R}^n$  and  $X$  is a standard normal vector, that is,  $X = (X_1, \dots, X_n)$  is jointly normally distributed with  $\mathbb{E}(X_j) = 0$  and  $\mathbb{E}(X_j X_k) = \delta_{jk}$ . It is a trivial observation that

$$\mathbb{E}(f(X)) = \mathbb{E}(f(UX)) \quad (1)$$

for every orthogonal transform  $U : \mathbb{R}^n \rightarrow \mathbb{R}^n$ . It has been observed in a number of articles ([1, 13, 11]) that, while this reformulation does not change the simulation problem from the probabilistic point of view, it does make a – sometimes big – difference when quasi-Monte Carlo (QMC) simulation is applied to generate the realizations of  $X$ .

Prominent examples are supplied by the well-known Brownian bridge [13] and principal component analysis (PCA) [1] constructions of Brownian paths which will be detailed in the following paragraphs. Assume we want to calculate an approximation to  $\mathbb{E}(g(B))$  where  $B$  is a Brownian motion with index set  $[0, T]$ . In most applications this can be reasonably approximated by  $\mathbb{E}(\tilde{g}(B_{\frac{T}{n}}, \dots, B_{\frac{Tn}{n}}))$ , where  $\tilde{g}$  is a corresponding function taking as its argument a *discrete Brownian path*, by which we mean a normal vector with

---

\*Christian Irrgeher, Institute of Financial Mathematics, Johannes Kepler University Linz, Altenbergerstraße 69, A-4040 Linz, Austria. e-mail: christian.irrgeher@jku.at The author is supported by the Austrian Science Foundation (FWF), Project P21943.

†Gunther Leobacher, Institute of Financial Mathematics, Johannes Kepler University Linz, Altenbergerstraße 69, A-4040 Linz, Austria. e-mail: gunther.leobacher@jku.at The author is partially supported by the Austrian Science Foundation (FWF), Project P21196.

covariance matrix

$$\Sigma := \left( \frac{T}{n} \min(j, k) \right)_{j,k=1}^n = \frac{T}{n} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 2 & \dots & 2 \\ 1 & 2 & 3 & \dots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 3 & \dots & n \end{pmatrix}.$$

There are three classical methods for sampling from  $(B_{\frac{T}{n}}, \dots, B_{\frac{nT}{n}})$  given a standard normal vector  $X$ , namely the forward method, the Brownian bridge construction and the principal component analysis construction. All of these constructions may be written in the form  $(B_{\frac{T}{n}}, \dots, B_{\frac{nT}{n}}) = AX$ , where  $A$  is an  $n \times n$  real matrix with  $AA^\top = \Sigma$ .

For example, the matrix corresponding to the forward method is

$$A = S := \sqrt{\frac{T}{n}} \begin{pmatrix} 1 & 0 & \dots & 0 \\ 1 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}, \quad (2)$$

while PCA corresponds to  $A = VD$ , where  $\Sigma = VD^2V^\top$  is the singular value decomposition of  $\Sigma$ . A corresponding decomposition for the Brownian bridge algorithm is given, for example, in Larcher, Leobacher & Scheicher [10].

It has been observed by Papageorgiou [15] that  $AA^\top = \Sigma$  if and only if  $A = SU$  for some orthogonal matrix  $U$ , so that every linear construction of  $(B_{\frac{T}{n}}, \dots, B_{\frac{nT}{n}})$  corresponds to an orthogonal transform of  $\mathbb{R}^n$ . In that sense the forward method corresponds to the identity, PCA corresponds to  $S^{-1}VD$  and Brownian bridge corresponds to the inverse Haar transform, see Leobacher [12].

Thus our original simulation problem can be written, as

$$\mathbb{E}(\tilde{g}(B_{\frac{T}{n}}, \dots, B_{\frac{nT}{n}})) = \mathbb{E}(\tilde{g}(SX)) = \mathbb{E}(f(X))$$

with  $f = \tilde{g} \circ S$ . In the context of discrete Brownian paths this corresponds to the forward method. Consequently, the same problem using the Brownian bridge takes on the form  $\mathbb{E}(f(H^{-1}X))$ , where  $H$  is the matrix of the Haar transform, and has the form  $\mathbb{E}(f(S^{-1}VDX))$ , with  $S, V, D$  as above, when PCA is used.

Papageorgiou [15] noted that whether or not the Brownian bridge and PCA constructions enhance the performance of QMC methods depends critically on the integrand  $f$  and he provides an example of a financial option where those two methods give much worse results than the forward method. That lead to the idea of searching for orthogonal transform tailored to the integrand. Imai & Tan [8] propose a general technique for this problem which they call linear transform (LT) method.

The exact reason why orthogonal transforms might have the effect to make a problem more suitable for QMC is still unknown. Caffish et. al. [3] propose that those transforms diminish the so-called *effective dimension* of the problem. Owen [14] provided the concept of *effective dimension of a function space*. The least that can be said with confidence is that introducing an orthogonal transform does not introduce a bias and that there are

choices (like the identity) that make the problem at least equally well suited for QMC as the original one.

While applying a suitable orthogonal transform to an integration problem may increase the performance of QMC simulation, there is also a disadvantage: the computation of the orthogonal transform incurs a cost, which in general is of the order  $O(n^2)$ . For large  $n$  this cost is likely to swallow any gains from the transform. In [12] it is therefore proposed to concentrate on orthogonal transforms which have cost of the order  $O(n \log(n))$  or less.

Examples of such *fast orthogonal transforms* include discrete sine and cosine transform, Walsh and (inverse) Haar transform as well as the orthogonal matrix corresponding to the PCA, see Scheicher [16] and Leobacher [12].

A relatively recent approach to enhance the efficiency of Monte Carlo simulation has been proposed by Giles [4] and Heinrich [7]. They propose a multilevel procedure by combining Monte Carlo based on different time discretizations. The improvement in computational efficiency by using quasi-Monte Carlo instead of Monte Carlo together with the multilevel method is shown in Giles & Waterhouse [5] where the authors used a rank-1 lattice rule with a random shift. Furthermore, they give a short discussion on the three classical sampling methods mentioned above. We contribute to the topic by finding an orthogonal transform adapted to the multilevel method, thus making it even more efficient.

The remainder of the paper is organized as follows. Section 2 reviews basic properties of Householder reflections and in Section 3 we describe an algorithm for finding a fast orthogonal transform using Householder reflections. The main part of our article, Section 4, recalls some of the basics of multilevel (quasi-)Monte Carlo and discusses how the ideas of Section 3 can be carried over to multilevel quasi-Monte Carlo integration.

Section 5 gives a numerical example where the method described earlier is applied to an example from finance. We will see that the method improves the efficiency of multilevel quasi-Monte Carlo integration.

## 2 Householder Reflections

We recall the definition and basic properties of Householder reflections from Golub & Van Loan [6].

**Definition 2.1.** *A matrix of the form*

$$U = I - 2 \frac{vv^\top}{v^\top v},$$

*where  $v \in \mathbb{R}^n$ , is called a Householder reflection. The vector  $v$  is called the defining Householder vector.*

In the following proposition,  $e_1$  denotes the first canonical basis vector in  $\mathbb{R}^n$ ,  $e_1 = (1, 0, \dots, 0)$ .

**Proposition 2.2.** *Householder reflections have the following properties:*

1. Let  $U$  be a Householder reflection with Householder vector  $v$ . If  $x \in \mathbb{R}^n$  is a vector then  $Ux$  is the reflection of  $x$  in the hyperplane  $\text{span}\{v\}^\perp$ . In particular,  $U$  is orthogonal and symmetric, i.e.  $U^{-1} = U$ .
2. Given any vector  $a \in \mathbb{R}^n$  we can find  $v \in \mathbb{R}^n$  such that for the corresponding Householder reflection  $U$  we have  $Ua = \|a\|e_1$ . The computation of the Householder vector uses  $3n$  floating point operations.
3. The computation of  $Ux$  uses at most  $4n$  floating point operations.

*Proof.* See Golub & Van Loan [6, Chapter 5.1]. □

### 3 Regression Algorithm

In this section we give a short description of a rather general method for constructing fast and efficient orthogonal transforms. Parts of the material have already been presented in [9], but we include them to make the paper self-contained.

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be a measurable function with  $\mathbb{E}(f(X)^2) < \infty$  for a standard normal vector  $X$ . Wang & Sloan [17] consider functions of the form

$$f(X) = g(w_1^\top X, \dots, w_m^\top X) \quad (3)$$

with  $w_1, \dots, w_m \in \mathbb{R}^n$  and  $g : \mathbb{R}^m \rightarrow \mathbb{R}$ . The authors show that for such functions there exists an orthogonal transform that reduces the dimension of  $f$  to at most  $m$ . Therefore the integration problem is not as high-dimensional as it seems and we have a convergence rate of the QMC algorithm applied to the transformed problem corresponding to  $m$  rather than  $n$ . We give a slightly modified version of their arguments to reduce the dimension of  $f$ , because we suggest using Householder reflections to generate the orthogonal transform which guarantees that the transform can be applied using at most  $O(n \log(n))$  operations if  $m \leq \log(n)$ .

Assume that  $w_1$  is not the zero vector and let  $U_1 : \mathbb{R}^n \rightarrow \mathbb{R}^n$  be a Householder reflection which maps  $e_1$  to  $w_1/\|w_1\|$ . Then  $w_1^\top U_1 X = \|w_1\|e_1^\top X = \|w_1\|X_1$  and therefore

$$f(U_1 X) = g(\|w_1\|X_1, (U_1 w_2)^\top X, \dots, (U_1 w_m)^\top X).$$

Next we write  $(U_1 w_k)^\top X = (U_1 w_k)_1^\top X_1 + (U_1 w_k)_{2..n}^\top X_{2..n}$ . That is,

$$f(U_1 X) = g_1(X_1, w_{2,1}^\top X_{2..n}, \dots, w_{m,1}^\top X_{2..n})$$

where  $w_{2,1}, \dots, w_{m,1} \in \mathbb{R}^{n-1}$ . Assuming that  $w_{2,1} \neq 0$ , let  $\tilde{U}_2 : \mathbb{R}^{n-1} \rightarrow \mathbb{R}^{n-1}$  be the Householder reflection that maps  $e_1$  to  $w_{2,1}/\|w_{2,1}\|$  and let

$$U_2 = \begin{pmatrix} 1 & 0 \\ 0 & \tilde{U}_2 \end{pmatrix}.$$

Then  $U_2$  is a Householder reflection from  $\mathbb{R}^n$  to  $\mathbb{R}^n$  and

$$f(U_1 U_2 X) = g_2(X_1, X_2, w_{3,2}^\top X_{3..n}, \dots, w_{m,2}^\top X_{3..n})$$

with  $w_{3,2}, \dots, w_{m,2} \in \mathbb{R}^{n-2}$ . Proceeding that way one arrives at

$$f(U_1 \cdots U_{\hat{m}} X) = g_{\hat{m}}(X_1, X_2, \dots, X_{\hat{m}})$$

for some  $\hat{m} \leq m$  (We may have  $\hat{m} < m$  if some transformed  $w_k$  are zero).

In the spirit of [17] we propose a procedure for more general integration problems. Let us assume that the function  $f$  is of the form

$$f(x) = \tilde{g}(h_1(x), \dots, h_m(x))$$

where  $m < n$ ,  $\tilde{g} : \mathbb{R}^m \rightarrow \mathbb{R}$  and  $h_k : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $k = 1, \dots, m$ . We want to approximate every  $h_k$  by a linear function, i.e.

$$h_k(x) \approx a_k^\top x + b_k$$

with  $a_k \in \mathbb{R}^n$  and  $b_k \in \mathbb{R}$ . The approximation is done by a “linear regression” approach and therefore, for every  $k = 1, \dots, m$ , we minimize

$$\mathbb{E} \left( (h_k(X) - a_k^\top X - b_k)^2 \right) \rightarrow \min.$$

First order conditions give for  $k = 1, \dots, m$

$$a_{k,j} = \mathbb{E}(X_j h_k(X)), \quad j = 1, \dots, n; \quad (4)$$

$$b_k = \mathbb{E}(h_k(X)). \quad (5)$$

Therefore, (4)-(5) minimizes the variance of the difference between each  $h_k(X)$  and its linear approximation  $a_k^\top X + b_k$ . So

$$\begin{aligned} \mathbb{V}(h_k(X)) &= \mathbb{E} \left( (h_k(X) - b_k)^2 \right) \\ &= \mathbb{E} \left( (a_k^\top X)^2 + (h_k(X) - b_k - a_k^\top X)^2 \right) \\ &= \|a_k\|^2 + \mathbb{V}(h_k(X) - a_k^\top X). \end{aligned}$$

That is,  $\|a_k\|^2 / \mathbb{V}(h_k(X))$  measures the fraction of variance captured by the linear approximation.

Now we approximate the function  $f$  by substituting the  $h_k$  with the linear functions obtained by the linear regression, i.e.

$$f(x) \approx \tilde{g}(a_1^\top x + b_1, \dots, a_m^\top x + b_m) = g(a_1^\top x, \dots, a_m^\top x).$$

Therefore  $f$  is approximated by a function of the form (3) and we can proceed in the same way as at the beginning of this section to determine a fast orthogonal transform by using Householder reflections.

Note that the method is only practical if the expectations  $\mathbb{E}(X_j h_k(X))$  in (4) can be computed explicitly or at least efficiently. After the statement of the algorithm we will give an example where explicit calculation is possible.

**Algorithm 3.1.** *Let  $X_1, \dots, X_n$  be independent standard normal variables. Let  $f$  be a function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ , which is of the form  $f = g \circ h$  where  $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$  and  $g : \mathbb{R}^m \rightarrow \mathbb{R}$ .*

1. Start with  $k, \ell = 1$  and  $U = I$ ;
2.  $a_{k,j} := \mathbb{E}(X_j h_k(UX))$  for  $j = k, \dots, n$ ;
3.  $a_{k,j} := 0$  for  $j = 1, \dots, k-1$ ;
4. if  $\|a_k\| = 0$  go to 7;
5. else let  $U_\ell$  be a Householder reflection that maps  $e_\ell$  to  $a_k/\|a_k\|$ ;
6.  $U = UU_\ell$ ;  $\ell = \ell + 1$ ;
7.  $k = k + 1$ ;
8. while  $k \leq m$ , go back to 2;
9. Compute  $\mathbb{E}(f(UX))$  using QMC.

*Example 3.2.* We give an example from finance for which Algorithm 3.1 can be applied efficiently. Motivated by a discrete arithmetic Asian option let us consider

$$f(X) = \max \left( \sum_{k=1}^n w_k \exp \left( \sum_{i=1}^n (c_{k,i} X_i + d_{k,i}) \right) - K, 0 \right).$$

with  $w_k, c_{k,i}, d_{k,i} \in \mathbb{R}$ . Now we can write  $f(X) = g(h_1(X))$  with  $g(y) = \max(y - K, 0)$  and  $h_1(X) = \sum_{k=1}^n w_k \exp(\sum_{i=1}^n (c_{k,i} X_i + d_{k,i}))$ . In that case we can compute  $\mathbb{E}(X_j h_1(X))$  explicitly. It is easily verified that, with  $\phi$  denoting the standard normal density,  $\phi(x) = \exp(-x^2/2)/\sqrt{2\pi}$ ,

$$\int_{\mathbb{R}} \exp(cx + d) \phi(x) dx = \exp(c^2/2 + d)$$

and

$$\int_{\mathbb{R}} x \exp(cx + d) \phi(x) dx = c \exp(c^2/2 + d)$$

for any  $c, d \in \mathbb{R}$ . Therefore, we obtain

$$\begin{aligned} a_{1,j} &= \mathbb{E}(X_j h_1(X)) \\ &= \int_{\mathbb{R}} \dots \int_{\mathbb{R}} x_j h_1(x) \phi(x_1) \dots \phi(x_n) dx_1 \dots dx_n \\ &= \sum_{k=1}^n w_k c_{k,j} \exp \left( \sum_{i=1}^n \frac{c_{k,i}^2}{2} + d_{k,i} \right). \end{aligned}$$

In [9] it was calculated that for practical parameters  $\|a\|^2$  is typically larger than  $0.99 \cdot \mathbb{V}(h_1(X))$ .

## 4 Multilevel Quasi-Monte Carlo

We start with an abstract formulation of the multilevel (quasi-)Monte Carlo method: suppose we want to approximate  $\mathbb{E}(Y)$  for some random variable  $Y$  which has finite expectation. Suppose further that we have a sequence of sufficiently regular functions  $f^\ell : \mathbb{R}^{m^\ell} \rightarrow \mathbb{R}$  such that

$$\lim_{\ell \rightarrow \infty} \mathbb{E}(f^\ell(X^\ell)) = \mathbb{E}(Y), \quad (6)$$

where for each  $\ell \geq 0$ ,  $X^\ell$  denotes an  $m^\ell$ -dimensional standard normal vector. (6) states that there exists a sequence of algorithms which approximate  $\mathbb{E}(Y)$  with increasing accuracy. For example, if  $f^\ell(X^\ell)$  has finite variance, we can approximate  $\mathbb{E}(Y)$  by  $\frac{1}{N} \sum_{k=0}^{N-1} f^\ell(X_k^\ell)$  using sufficiently large  $\ell$  and  $N$ , where  $(X_k^\ell)_{k \geq 0}$  is a sequence of independent standard normal vectors.

Usually, evaluation of  $f^\ell(X^\ell)$  becomes more costly with increasing  $\ell$  and  $N$ . Multilevel methods sometimes help us to save significant proportions of computing time by computing more samples for the coarser approximations, which need less computing time but have higher variance.

We will need the following definition in the statement of the multilevel Monte Carlo method: for any  $m \in \mathbb{N}$  and any  $\ell \in \mathbb{N}_0$  we call the  $m^{\ell-1} \times m^\ell$  matrix  $C_{m,\ell} = ((C_{m,\ell})_{i,j})_{i,j}$  with

$$(C_{m,\ell})_{i,j} := \begin{cases} \frac{1}{\sqrt{m}} & \text{if } (i-1)m + 1 \leq j \leq im \\ 0 & \text{else} \end{cases}$$

the *coarsening matrix* from level  $\ell$  to level  $\ell - 1$ . For example, in the case of  $m = 2$  the coarsening matrix is given by

$$C_{2,\ell} := \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}.$$

The following lemma is simple to verify and therefore we leave the proof to the reader.

**Lemma 4.1.** *Let  $m \in \mathbb{N}$ . If  $X^\ell$  is an  $m^\ell$ -dimensional standard normal vector, then  $C_{m,\ell}X^\ell$  is an  $m^{\ell-1}$ -dimensional standard normal vector.*

□

Obviously we have

$$\begin{aligned} \mathbb{E}(Y) &\approx \mathbb{E}(f^L(X^L)) = \mathbb{E}(f^0(X^0)) + \sum_{\ell=1}^L \mathbb{E}(f^\ell(X^\ell)) - \mathbb{E}(f^{\ell-1}(X^{\ell-1})) \\ &= \mathbb{E}(f^0(X^0)) + \sum_{\ell=1}^L \mathbb{E}(f^\ell(X^\ell)) - \mathbb{E}(f^{\ell-1}(C_{m,\ell}X^\ell)) \\ &= \mathbb{E}(f^0(X^0)) + \sum_{\ell=1}^L \mathbb{E}(f^\ell(X^\ell) - f^{\ell-1}(C_{m,\ell}X^\ell)) \end{aligned} \quad (7)$$

Equation (7) becomes useful if, as is often the case in practice, the expectation  $\mathbb{E}(f^\ell(X^\ell) - f^{\ell-1}(C_{m,\ell}X^\ell))$  can be approximated to the required level of accuracy using less function evaluations for bigger  $\ell$  while the costs per function evaluation increases. One typical situation where this occurs is when a stochastic differential equation is solved numerically using time discretization with  $m^\ell$  time steps and  $f^\ell$  is some function on the set of solution paths. See [4] for how to exploit this representation.

In finance,  $f^\ell$  is typically of the form  $f^\ell(X) = \psi(h^\ell(X))$  for some functions  $h^\ell : \mathbb{R}^{m^\ell} \rightarrow \mathbb{R}$  and  $\psi : \mathbb{R} \rightarrow \mathbb{R}$ . In that context,  $h^\ell$  is some function taking as its argument a discrete (geometric) Brownian path, like the maximum or the average, and  $\psi$  is the payoff that depends on the outcome of  $h^\ell$ .

$$\begin{aligned}\mathbb{E}(f^L(X^L)) &= \mathbb{E}(f^0(X^0)) + \sum_{\ell=1}^L \mathbb{E}(\psi(h^\ell(X^\ell)) - \psi(h^{\ell-1}(C_{m,\ell}X^\ell))) \\ &= \mathbb{E}(\psi(h^0(X^0))) + \sum_{\ell=1}^L \mathbb{E}(g^\ell(h_1^\ell(X^\ell), h_2^\ell(X^\ell))) ,\end{aligned}$$

where  $h_1^\ell = h^\ell$ ,  $h_2^\ell = h^{\ell-1} \circ C_{m,\ell}$  and  $g^\ell(y_1, y_2) = \psi(y_1) - \psi(y_2)$ .

Now the integrands are precisely of the form covered by Algorithm 3.1. It is therefore sensible to apply the corresponding orthogonal transform  $U^\ell : \mathbb{R}^\ell \rightarrow \mathbb{R}^\ell$  at each level such that we get

$$\mathbb{E}(f^L(X^L)) = \mathbb{E}(f^0(U^0 X^0)) + \sum_{\ell=1}^L \mathbb{E}(g^\ell(h_1^\ell(U^\ell X^\ell), h_2^\ell(U^\ell X^\ell))) .$$

Of course, we are free to try any other set of orthogonal transforms, like PCA. The advantage of using the regression algorithm is that here at each level the orthogonal transform is determined by taking both the fine and the coarse discretization into account.

In the next section we shall try our method on a concrete example from finance, the Asian option.

## 5 Asian Option

We will consider an Asian call option in the Black-Scholes model, i.e. under the risk-neutral measure the stock price process  $S = (S_t)_{t \geq 0}$  is given by the stochastic differential equation (SDE)

$$dS_t = rS_t dt + \sigma S_t dB_t$$

where  $r$  is the interest rate,  $\sigma$  is the volatility and  $(B_t)_{t \geq 0}$  is a standard Brownian motion. Given the stock price  $S_0$  at time 0, the solution of the SDE is given by

$$S_t = S_0 \exp\left(\left(r - \sigma^2/2\right)t + \sigma B_t\right) .$$

The payoff of the Asian call option with fixed strike price  $K$ , maturity  $T$  and underlying  $S$  is

$$\max\left(\frac{1}{T} \int_0^T S_t dt - K, 0\right) .$$

That is, if the average stock price over the time interval  $[0, T]$  is above level  $K$ , the option pays its holder at time  $T$  the difference between that value and  $K$ , otherwise it pays nothing.

Martingale pricing theory tells us that the price of the option is given by the discounted expectation of the payoff function under the risk-neutral measure, see Björk [2, Chapter 10], i.e.

$$C = \exp(-rT) \mathbb{E} \left( \max \left( \frac{1}{T} \int_0^T S_t dt - K, 0 \right) \right).$$

To approximate  $C$  in the time-continuous model, a common way is to use (multilevel) quasi-Monte Carlo integration to compute the expectation. To that end we first approximate the integral by a sum: For any equidistant time discretization with  $n \in \mathbb{N}$  points,

$$\frac{1}{T} \int_0^T S_t dt \approx \frac{1}{n} \sum_{k=1}^n S_k(X)$$

where  $X = (X_1, \dots, X_n)$  is a standard normal vector and

$$S_k(X) = S_0 \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{n} + \sigma \sqrt{\frac{T}{n}} \sum_{i=1}^k X_i \right), \quad k = 1, \dots, n.$$

Therefore the payoff function of the Asian option is approximately

$$f(X) = \max \left( \frac{1}{n} \sum_{k=1}^n S_k(X) - K, 0 \right). \quad (8)$$

If the time discretization consists of  $\ell = m^\ell$  points with  $\ell \in \mathbb{N}_0$ ,  $m \in \mathbb{N}$ , we denote the payoff function by  $f^\ell$  and it is therefore given by (8) with  $n = m^\ell$ .

Thus we can approximate the price  $C$  using multilevel QMC integration with finest level  $L$  by

$$\begin{aligned} C &\approx \exp(-rT) \left( \mathbb{E}(f^0(X^0)) + \sum_{k=1}^L \mathbb{E}(f^\ell(X^\ell) - f^{\ell-1}(C_{m,\ell} X^\ell)) \right) \\ &= \exp(-rT) \left( \mathbb{E}(f^0(X^0)) + \sum_{k=1}^L \mathbb{E}(g^\ell(h_1^\ell(X^\ell), h_2^\ell(X^\ell))) \right) \end{aligned}$$

where  $g^\ell(y_1, y_2) = \max(y_1, 0) - \max(y_2, 0)$ ,

$$h_1^\ell(X^\ell) = \frac{1}{m^\ell} \sum_{k=1}^{m^\ell} S_0 \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^\ell} + \sigma \sqrt{\frac{T}{m^\ell}} \sum_{i=1}^k X_i^\ell \right)$$

and

$$h_2^\ell(X^\ell) = \frac{1}{m^{\ell-1}} \sum_{k=1}^{m^{\ell-1}} S_0 \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^{\ell-1}} + \sigma \sqrt{\frac{T}{m^{\ell-1}}} \sum_{i=1}^k (C_{m,\ell} X^\ell)_i \right).$$

In applying Algorithm 3.1, we have to compute the vectors  $a_1^\ell, a_2^\ell$  for each level. This can be done as in Example 3.2. For  $\ell = 1, \dots, L$  and  $j = 1 \dots, m^\ell$  we get

$$\begin{aligned}
a_{1,j}^\ell &= \mathbb{E} \left( X_j h_1^{(\ell)} \right) \\
&= \sum_{k=1}^{m^\ell} \frac{S_0}{m^\ell} \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^\ell} \right) \mathbb{E} \left( X_j \exp \left( \sigma \sqrt{\frac{T}{m^\ell}} \sum_{i=1}^k X_i \right) \right) \\
&= \sum_{k=j}^{m^\ell} \frac{S_0}{m^\ell} \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^\ell} \right) \sigma \sqrt{\frac{T}{m^\ell}} \exp \left( \frac{\sigma^2}{2} \frac{T}{m^\ell} k \right) \\
&= \sum_{k=j}^{m^\ell} \frac{S_0 \sigma}{m^\ell} \sqrt{\frac{T}{m^\ell}} \exp \left( r k \frac{T}{m^\ell} \right)
\end{aligned}$$

and

$$\begin{aligned}
a_{2,j}^\ell &= \mathbb{E} \left( X_j h_2^{(\ell)} \right) \\
&= \sum_{k=1}^{m^{\ell-1}} \frac{S_0}{m^{\ell-1}} \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^{\ell-1}} \right) \mathbb{E} \left( X_j \exp \left( \sigma \sqrt{\frac{T}{m^\ell}} \sum_{i=1}^k \sum_{p=1}^m X_{(i-1)m+p} \right) \right) \\
&= \sum_{k=\lfloor \frac{j-1}{m} \rfloor + 1}^{m^{\ell-1}} \frac{S_0}{m^{\ell-1}} \exp \left( \left( r - \frac{\sigma^2}{2} \right) k \frac{T}{m^{\ell-1}} \right) \sigma \sqrt{\frac{T}{m^\ell}} \exp \left( \frac{\sigma^2}{2} \frac{T}{m^{\ell-1}} k \right) \\
&= \sum_{k=\lfloor \frac{j-1}{m} \rfloor + 1}^{m^{\ell-1}} \frac{S_0 \sigma}{m^{\ell-1}} \sqrt{\frac{T}{m^\ell}} \exp \left( r k \frac{T}{m^{\ell-1}} \right).
\end{aligned}$$

Now we compare the multilevel QMC method combined with the regression algorithm with multilevel Monte Carlo and multilevel quasi-Monte Carlo (forward and PCA sampling) numerically. For that we choose the parameters as  $r = 0.04$ ,  $\sigma = 0.3$ ,  $S_0 = 100$ ,  $K = 100$  and  $T = 1$ . At the finest level we start with  $2^{10}$  discretization points and at each coarser level we divide in half the number of points, i.e.  $L = 10$  and  $m = 2$ . Furthermore, the number of sample points are doubled at each level starting with  $N_L$  sample points at the finest level  $L$ . For the QMC approaches we take a Sobol sequence with a random shift. In Table 1 we compare for different values  $N_L$  both the average and the standard deviation of the price of the Asian call option based on 1000 independent runs. Moreover, the average computing time for one run is given in brackets. As we can see, the regression algorithm yields the lowest standard deviation, but the computing time of the regression algorithm is slightly worse than the forward method. However, the regression algorithm is better than the PCA construction measured in both standard deviation and computing time.

In Table 2 we compare the regression algorithm both for multilevel QMC and for QMC with  $2^{10}$  time steps ( $L = 10$ ). We can observe that the standard deviation as well as the computing time of the multilevel QMC setting is significantly better compared with crude QMC.

$N_L$	multilevel Monte Carlo		multilevel QMC					
	average	stddev	forward		PCA		regression	
	average	stddev	average	stddev	average	stddev	average	stddev
2	7.717 (0.0057 s)	$0.41 \times 10^0$	7.735 (0.0057 s)	$0.19 \times 10^{-1}$	7.736 (0.0088 s)	$0.16 \times 10^{-1}$	7.739 (0.0069 s)	$0.10 \times 10^{-1}$
4	7.738 (0.0074 s)	$0.19 \times 10^0$	7.734 (0.0074 s)	$0.71 \times 10^{-2}$	7.736 (0.0118 s)	$0.44 \times 10^{-2}$	7.738 (0.0091 s)	$0.29 \times 10^{-2}$
8	7.748 (0.0101 s)	$0.54 \times 10^{-1}$	7.737 (0.0100 s)	$0.30 \times 10^{-2}$	7.737 (0.0165 s)	$0.14 \times 10^{-2}$	7.736 (0.0124 s)	$0.10 \times 10^{-2}$
16	7.746 (0.0157 s)	$0.40 \times 10^{-1}$	7.736 (0.0157 s)	$0.11 \times 10^{-2}$	7.737 (0.0279 s)	$0.69 \times 10^{-3}$	7.736 (0.0194 s)	$0.30 \times 10^{-3}$
32	7.728 (0.0266 s)	$0.31 \times 10^{-1}$	7.736 (0.0265 s)	$0.49 \times 10^{-3}$	7.737 (0.0585 s)	$0.21 \times 10^{-3}$	7.736 (0.0326 s)	$0.10 \times 10^{-3}$
64	7.739 (0.0486 s)	$0.81 \times 10^{-2}$	7.736 (0.0484 s)	$0.20 \times 10^{-3}$	7.737 (0.1202 s)	$0.69 \times 10^{-4}$	7.737 (0.0583 s)	$0.32 \times 10^{-4}$

Table 1: Multilevel (Q)MC using  $2^{10}$  time steps ( $L = 10$ ). The average and the standard deviation of the option price are based on 1000 runs. The average computing time is given in brackets.

	average	stddev	time (s)
MLQMC - Regression ( $N_L = 2^6$ )	7.7366	$0.32 \times 10^{-4}$	0.0323
QMC - Regression ( $N = 2^{12}$ )	7.7362	$1.01 \times 10^{-4}$	0.1511

Table 2: QMC and multilevel QMC, both combined with the regression algorithm, with  $2^{10}$  time steps ( $L = 10$ ) based on 1000 runs.

## References

- [1] P. Acworth, M. Broadie, and P. Glasserman. A comparison of some Monte Carlo and quasi-Monte Carlo techniques for option pricing. In H. Niederreiter, P. Hellekalek, G. Larcher, and P. Zinterhof, editors, *Monte Carlo and Quasi-Monte Carlo Methods 1996, Proceedings of a Conference at the University of Salzburg, Austria, July 9–12, 1996*, pages 1–18, New York, 1998. Springer.
- [2] T. Björk. *Arbitrage Theory in Continuous Time*. Oxford University Press, New York, third edition, 2009.
- [3] R. Caflisch, W. Morokoff, and A. Owen. Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. *Journal of Computational Finance*, 1(1):27–46, 1997.
- [4] M. B. Giles. Multilevel Monte Carlo path simulation. *Oper. Res.*, 56(3):607–617, 2008.
- [5] M. B. Giles and B. J. Waterhouse. Multilevel quasi-Monte Carlo path simulation. Albrecher, Hansjörg (ed.) et al., *Advanced financial modelling*. Berlin: Walter

- de Gruyter. Radon Series on Computational and Applied Mathematics 8, 165-181 (2009)., 2009.
- [6] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, third edition, 1996.
  - [7] S. Heinrich. Multilevel Monte Carlo methods. Margenov, Svetozar (ed.) et al., Large-scale scientific computing. 3rd international conference, LSSC 2001, Sozopol, Bulgaria, June 6-10, 2001. Revised papers. Berlin: Springer. Lect. Notes Comput. Sci. 2179, 58-67 (2001)., 2001.
  - [8] J. Imai and K. S. Tan. A general dimension reduction technique for derivative pricing. *J. Comput. Finance*, 10:129–155, 2007.
  - [9] C. Irrgeher and G. Leobacher. Fast orthogonal transforms for pricing derivatives with quasi-Monte Carlo. In C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, editors, *Proceedings of the 2012 Winter Simulation Conference*, 2012. To appear.
  - [10] G. Larcher, G. Leobacher, and K. Scheicher. On the tractability of the Brownian bridge algorithm. *J. Complexity*, 19:511–528, 2003.
  - [11] G. Leobacher. Stratified sampling and quasi-Monte Carlo simulation of Lévy processes. *Monte-Carlo methods and applications*, 12(3-4):231–238, 2006.
  - [12] G. Leobacher. Fast orthogonal transforms and generation of Brownian paths. *J. Complexity*, 28:278–302, 2012.
  - [13] B. Moskowitz and R. E. Caflisch. Smoothness and dimension reduction in Quasi-Monte Carlo methods. *Math. Comput. Model.*, 23(8-9):37 – 54, 1996.
  - [14] A. B. Owen. Effective dimension for weighted function spaces. Technical report, Department of Statistics, Stanford University, 2012.
  - [15] A. Papageorgiou. The Brownian bridge does not offer a consistent advantage in quasi-Monte Carlo integration. *J. Complexity*, 18(1):171–186, 2002.
  - [16] K. Scheicher. Complexity and effective dimension of discrete Lévy areas. *J. Complexity*, 23(2):152–168, 2007.
  - [17] I. H. Sloan and X. Wang. Quasi-Monte Carlo methods in financial engineering: An equivalence principle and dimension reduction. *Operations Research*, 59(1):80–95, 2011.